A Software-defined Delay-aware Traffic Load Control for WiFi-based Smart City Services

Basima Kurungadan and Atef Abdrabou*
Department of Electrical Engineering, UAE University, Al-Ain, Abu Dhabi, 15551, UAE
Email:{201970069@uaeu.ac.ae, atef.abdrabou@uaeu.ac.ae}

2021 International Conference on Computer, Information and Telecommunication Systems (CITS) | 978-1-6654-4913-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/CITS52676.2021.9617928

***Abstract*—Most Smart city initiatives embrace WiFi as the candidate technology for its ubiquitous coverage and cost-effectiveness. Nevertheless, the quality-of-service (QoS) requirements of different applications vary, hindering the applicability of some smart city services. This paper addresses the usage of software-defined n etworking (SDN) t o c ontrol t he t raffic load of WiFi access points constituting a city-wide coverage. The traffic load control aims to satisfy the packet delay requirements of smart city applications without the need of having special requirements on the WiFi nodes such as running a software agent, sending probe packets, or supporting specific roaming protocol. A traffic load control algorithm is developed and tested using hardware experiments performed in a practical setting that mimics the actual scenario. The performance results show that the developed algorithm is effective in maintaining low packet transfer delay.**

Keywords – Software-defined N etworking, W iFi, traffic, load, control, delay, IoT, smart city, e-Health.

I. Introduction

According to a recent forecast, 71% of IP traffic w ill be from mobile and WiFi-connected devices. This is supposed to accompany a rise in the number of public hotspots to 549 million by 2022 [1]. Moreover, the Middle East and Africa have the highest mobile data traffic growth of any region with a 41% CAGR (Compound Annual Growth Rate) according to [1]. In addition, wearable devices are getting popular, and instead of embedded cellular connectivity, they are now being connected through smartphones [2]. There is also an increased demand for uninterrupted video streaming, for which WiFi offloading has been of great help [3]. Moreover, the existence of the COVID-19 pandemic extensively increased the demand for using the Internet via WiFi connections for many purposes that require high speed in both upload and download, such as online business meetings (video conferencing) and online education. Furthermore, with the vast number of hotspots currently being established in public, many smart city initiatives adopt the usage of WiFi to realize a multitude of Internet of Things (IoT) data services [4]. Although the smart city is not a new concept, its development is still in the nascent phase. People are still hesitant to trust most of its capabilities due to various reasons such as security, privacy, reliability, and availability [5]. Indeed, transferring a huge amount of real-time data from IoT devices, the analysis of this data, and taking proper actions are the primary functions involved in most of the smart city applications [6]. Some of the actions are time-critical, and hence any delay in data transfer would result in severe consequences. For example, smart city e-health applications are essential for both the elderly and younger citizens, but data transfer delay is critical for these applications to perform their intended functions, especially in emergencies [7]. In many smart city initiatives, the city is covered by many WiFi networks with overlapped coverage. This may result in a scenario where some WiFi networks are loaded with a routine data collection, while other delay-sensitive traffic (e.g., reporting an emergency alert of a person's health) needs to be sent within a short time. Thus, the careful radio resource management of smart city WiFi networks is inevitable for smart city services to achieve their targets. This requires a dynamic configuration of network resources, which cannot be achieved by conventional methods. SDN is a growing paradigm that has great potential in addressing networking issues that require flexible configuration, and resource control [8]. In this paradigm, the network data plane and control plane are separated, which paves the way to control a network by software functions from a central controller without the involvement of many middleboxes. This paper uses the SDN concept to control data traffic delay in a smart city wireless network consisting of several WiFi networks with overlapped coverage. The SDN controller decides which access point (AP) the wireless node will be connected to, as opposed to the case where the node itself makes this decision in traditional networks [9]. Moreover, the decision is not based on received signal strength indicator (RSSI) values, which may result in uneven load distribution among different access points [10]. Furthermore, the wireless nodes do not probe the network by sending messages to different APs. Instead, the controller continuously monitors the network's state and redistributes the clients over different access points based on their end-to-end (E2E) delay requirements. Although some recent amendments to the IEEE 802.11 standards (IEEE 802.11k/v) [11][12] provide signaling messages for the WiFi nodes to change their AP attachments, these new standards usually are not implemented over IoT wireless sensor nodes since they are designed to be simple with a low cost.

The paper makes the following contributions. First, it devises an algorithm for determining the smart city WiFi node with the largest packet transfer delay in a non-invasive fashion (i.e., without installing an agent on the node hardware or measuring the E2E delay at receiving nodes). Second, the controller handovers the client to a less loaded AP seamlessly without changing the node's network configuration or sending handover messages. The operation and efficacy of the algorithm have been measured experimentally in a lab setting that

*Corresponding author.

mimics a real scenario.

The rest of the paper is organized as follows. Section II reviews the most relevant research works in the literature. In Section III, the system model is outlined. Section IV introduces the proposed algorithm. The details of the experimental setup are described in Section V. Section VI presents the experimental results and provides a discussion about the findings. The paper is concluded in Section VII.

II. Related Works

Several research works in the literature target WiFi AP selection. However, the majority of these works require a special arrangement in the WiFi client such as sending probe packets [13] [14], running a software agent [15] [16], or requires a specific protocol support [17]. For instance, an online algorithm is presented in [13]. In this algorithm, a WiFi client selects an AP within its transmission range such that the norm of the other APs' loads is minimized. However, the algorithm targets only data throughput and needs to change the firmware of the WiFi adapter to be able to send special probe packets. Also, in [15], a system named Virgil is proposed. It selects APs after performing a batch of tests by quickly associating with all nearby APs to assess the quality of their Internet connections. In [17], a home AP and range extender selection are discussed. Although the proposed work in [17] considers the channel load, instead of the RSSI values only, it requires the availability of IEEE 802.11k/v support in WiFi adapters.

On the other hand, some research works propose using SDN in AP selection. The authors of [18] propose an AP selection algorithm that runs on a centralized SDN controller based on a fittingness factor depending on bit rate without considering packet delay. In [19], the authors introduce a performance analysis of a simulated software defined-WiFi network considering latency and packet loss for e-health applications. However, the works in [18] and [19] are tested only using computer simulations. To the best of our knowledge, no other research work in the literature introduces experimentally-validated delay-aware traffic load control scheme for overlapped WiFi networks using SDN with continuous monitoring of WiFi client traffic and seamless handover without adding any special capabilities to the WiFi client adapters, which suits the simplicity required for IoT/smart city devices/applications.

III. System Model

Consider a smart city scenario where the city is covered by a large number of WiFi networks with overlapped coverage. Here, we assume the existence of different smart city applications as shown in 1. Some of these applications are delay-sensitive such as eHealth and video surveillance. Other smart city applications that do not have strict real-time delay requirements, such as IoT-connected vehicles for fleet management or smart parking, are also considered. All the wireless nodes (clients) are assumed to run the legacy WiFi IEEE 802.11 a/g/n standards since IoT nodes are normally low in cost as they are deployed in large volumes for different smart city applications, and hence they do not support the IEEE 802.11 v/k standard amendments.

An SDN controller is assumed to be able to connect to all the APs in the network. The controller is assumed to oversee the traffic information of every WiFi node (client) connected to the network and the delay requirements of such nodes. The controller can also hand over a WiFi node to another AP that covers the node. For simplicity, we assume that the WiFi nodes are stationary or slowly moving in the area where they are covered by multiple APs. The handover is supposed to be seamless (i.e., it neither needs over-the-air reconfiguration nor causes connection disruption).

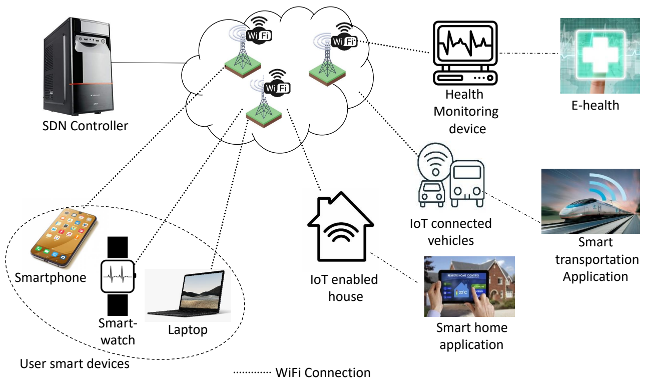


Fig. 1. System Model

IV. Analysis and Proposed Algorithm

The end-to-end delay, D_j, for a packet of a node j, is given by the sum of its service time, S_{t_j}, and the queueing delay, Q_{t_j}, at the sender.

$$D_j = S_{t_j} + Q_{t_j}. \tag{1}$$

Generally, Q_{t_j} depends on the arrival process in addition to the service process, which also controls the service time. For simplicity, we assume that all nodes have a similar arrival process with nearly the same average rate α, and hence we use the variation of service time as a way of estimating the variation of the end-to-end delay (i.e., the node with a long service time will also experience a large end-to-end delay). In IEEE 802.11, S_{t_j} can be generalized from [20] as

$$S_{t_j} = \sum_{i=1}^{N-1} \gamma_i \left[T_{s_i} + \frac{T_{c_i}}{2} \frac{p}{1-p} \right] + B(p) + T_{s_j} + \frac{T_{c_j}}{2} \frac{p}{1-p} \tag{2}$$

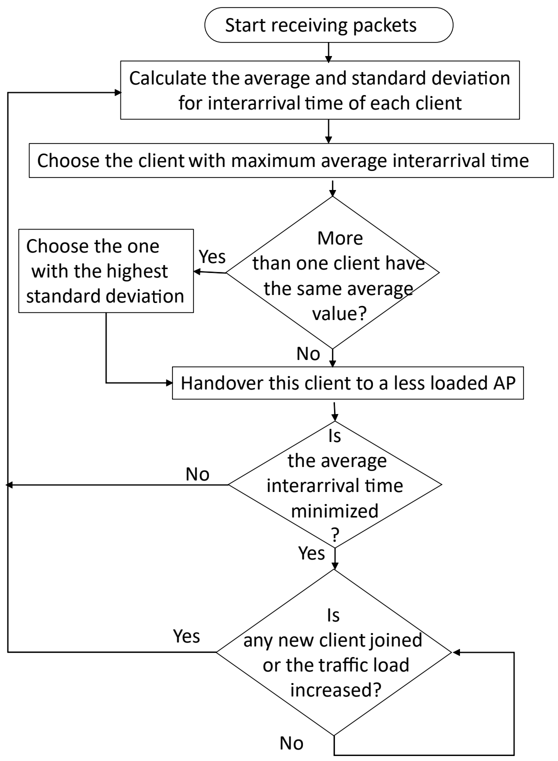
where $\gamma_i = S_{t_i}\alpha$ is the queue utilization factor of node i, α is the packet arrival rate, p is the collision probability (assumed almost fixed for all nodes), and B(p) is the average backoff period as given in eq. (25) in [20]. T_{s_i} is the packet transmission time for node i, and T_{c_i} is the packet collision time for node i. They can be expressed as $T_{s_i} = \frac{L}{R_i} + T_{M_1}$ and $T_{c_i} = \frac{L}{R_i} + T_{M_2}$, respectively, where T_{M_1} and T_{M_2} are constant times related to the IEEE 802.11 protocol operation [20]. The following steps outline the proposed algorithm.

Step 1: A packet-In event will be sent to the SDN controller when a packet is received at the Open vSwitch (OVS). The controller records the time of arrival of these events then calculates the average and standard deviation of the events' interarrival time, which correspond to the average and standard deviation, respectively, of the packet service time of each WiFi client.

Step 2: The client with the maximum average interarrival time is chosen for handover to a less loaded AP. If more than one client has obtained the same maximum average value, the one with the highest standard deviation will be chosen for handover.

Step 3: The SDN controller continues to monitor the Packet-In event interarrival time of the handed-over client to check if it is lowered. Otherwise, Step 2 will be repeated with another AP.

Step 4: The SDN controller also checks the status of the packet-In event interarrival time when a new client joins an AP or the traffic load increases (of a certain AP) since both affect the service time of the existing clients according to (2). Once the average interarrival time increases, Step 2 will be repeated, and so on. Figure 2 summarizes the algorithm in a flowchart.



Start receiving packets
Calculate the average and standard deviation for interarrival time of each client
Choose the client with maximum average interarrival time
More than one client have the same average value?
Yes
Choose the one with the highest standard deviation
No
Handover this client to a less loaded AP
Is the average interarrival time minimized ?
No
Yes
Is any new client joined or the traffic load increased?
Yes
No

Fig. 2. Proposed algorithm flow chart.

V. Experimental Setup

The experimental setup mimics the real-life scenario using hardware equipment running various software tools, as in the sequel.

The hardware equipment responsible for SDN functionality, data traffic generation, and reception consists of several Linux-based computers that run recent Ubuntu operating system versions (20.04 or 19.10). Two TP-LINK wireless dual-band gigabit routers are used as the access points AP1 and AP2. All the computers emulating WiFi nodes are equipped with IEEE 802.11g/n WiFi adapters, whereas the rest are connected using Gigabit Ethernet ports. Three sending WiFi nodes (referred hereafter as clients) generate data traffic to one receiving computer. Two other WiFi nodes are generating background traffic emulating the non-smart city traffic that may be concurrently transmitted. In addition, one computer acts as an SDN controller.

Different software tools are used to implement the SDN functionality as in the following. The SDN controller is implemented using Ryu [21]. The SDN controller, which works on a dedicated computer, uses the standard OpenFlow protocol messaging to communicate with another multi-Ethernet-ports computer running (OVS) software via an Ethernet connection, as shown in Fig. 3. This computer runs as a soft SDN switch. It connects the APs, SDN Controller, and receiving computer, which receives the data from the three WiFi clients. The APs' firmware also runs the OVS software. The OpenFlow protocol is not designed for wireless networks, and hence it can change data flow forwarding rules but cannot hand over WiFi nodes to other APs. For this reason, the Empower-5G framework [22] is used to perform the handover of WiFi clients. It runs on the same computer as the SDN controller, with an agent that runs on the APs.

The Empower-5G runtime controller creates a WiFi network with a unique service set identifier (SSID) at each AP. It performs the handover by using the light virtual access point (LVAP) agent. Whenever a WiFi node joins an AP, a corresponding LVAP agent is created in this AP. Consequently, a handover from one AP to another effectively means eliminating the LVAP from the former and creating a new LVAP in the latter.

The SDN controller runs the developed algorithm as an Ryu application, using the 5G-Empower framework to hand over the selected WiFi clients to other APs. All the data traffic is created using RUDE/CRUDE software tool, which is a client/server application generating UDP traffic. All the WiFi clients are synchronized to the receiving node using the precision time protocol (PTP) for accurate estimation of the end-to-end delay.

VI. Performance Results

A. Results

The performance of the proposed algorithm is extensively measured by laboratory experiments using three sets of performance indicators. For each set, different experiment samples are repeated around 30 times. The first set of results compares the average E2E delay of the WiFi clients before and after handover with varying client traffic rates and a constant background traffic load of 120 packets per second (for 1450-byte packets). As Fig. 4(a) reveals, the average packet delay after

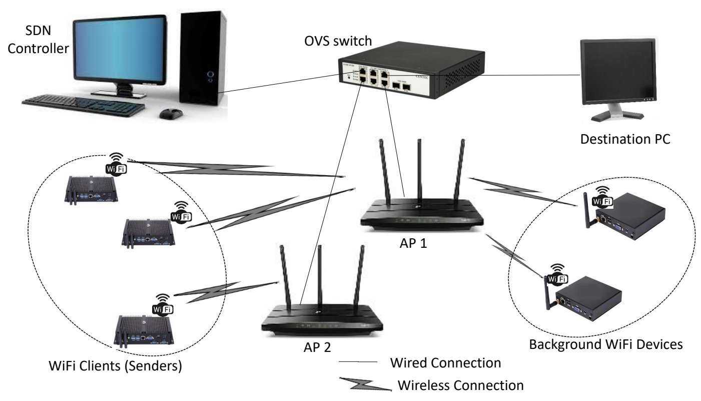


Fig. 3. Experimental Setup

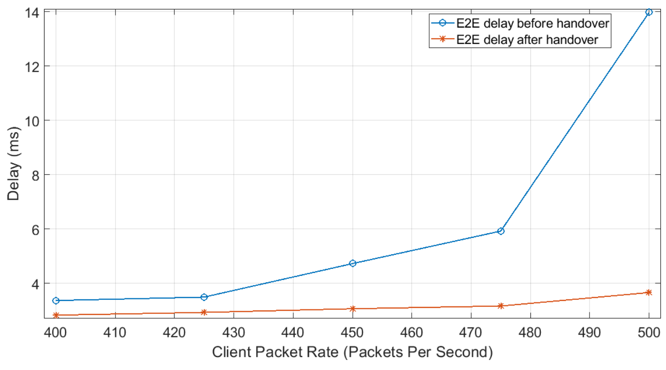
handover significantly decreased, especially for high client rates. The second set of results show that the variations of the packet-In event interarrival time at the Ryu controller reflect the change of the E2E delay as shown in Fig. 4(b) for a wide range of client rates. The third set of results compares the performance of the proposed handover algorithm with an ideal handover decision made by an independent observer who can measure the actual E2E delay at the receiving node. Fig. 4(c) reveals that the percentage in the number of times an ideal handover decision was different from the proposed algorithm decision decreases with increasing the client packet rate.

Moreover, for further validation of the proposed algorithm, experiments to test the same sets of performance indicators are conducted with varying the background traffic load while keeping the WiFi client data rate fixed at 475 packets per second. Fig. 5(a) reveals a significant difference in E2E delay before and after the handover is performed. In addition, Fig. 5(b) shows that the change of the E2E delay is generally reflected by the change of the packet interarrival time at the controller, except with high background traffic load, where a slight change in interarrival time is corresponding to a significant change in E2E delay. Again, Fig. 5 depicts that the percentage difference from an ideal handover decision decreases with increasing the background traffic load.

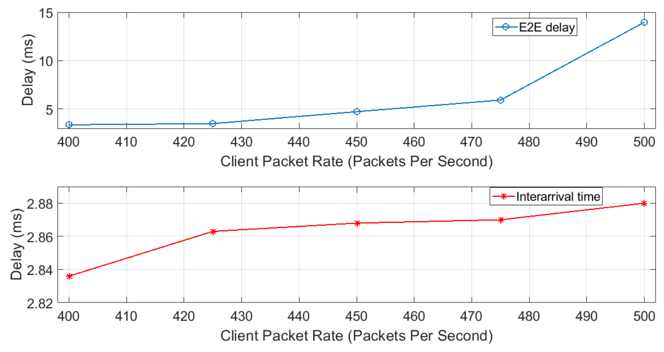
B. Discussion

The performance results introduced in Subsection VI-A indicate that using packet interarrival time at the SDN controller by the proposed algorithm to estimate the E2E delay and perform the handover is effective in reducing the packet delay for all the WiFi clients as revealed in Figs. 4(a) and 5(a), respectively.

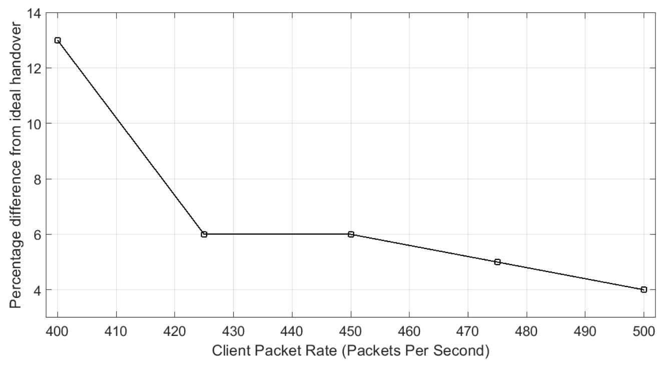
Although the variation of the packet-In event interarrival time (service time) is small when the packet rate of the clients is high, or the background traffic load is large (Figs. 4(b) and 5(b)), the proposed algorithm is able to select the client with the highest E2E delay for handover with a small error compared with low client rate or background traffic load. The reason for this trait is that high network load (high γ_i in (2)) leads to drive the network closer to saturation which lengthens



(a) E2E delay before and after handover of WiFi clients.



(b) A comparison of e2e delay and interarrival time at the SDN controller.



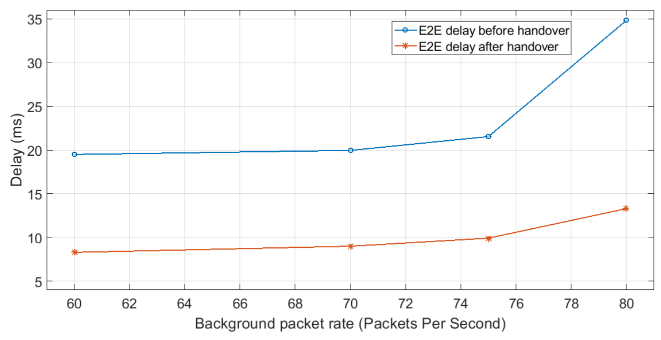
(c) Percentage difference between ideal handover and algorithm decision.

Fig. 4. Algorithm performance with different WiFi client data rates.

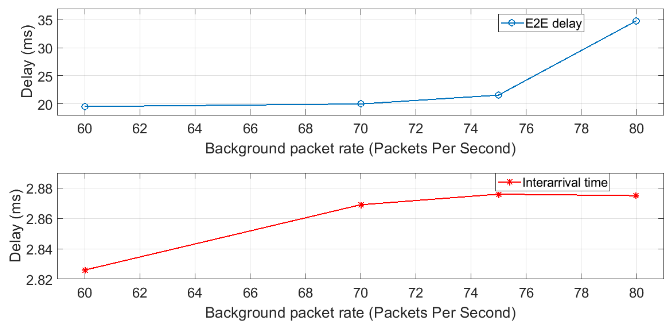
the service time and causes the queuing delay to be highly dependent on it (i.e., rapidly increases for a small variation of the service time).

VII. Conclusion

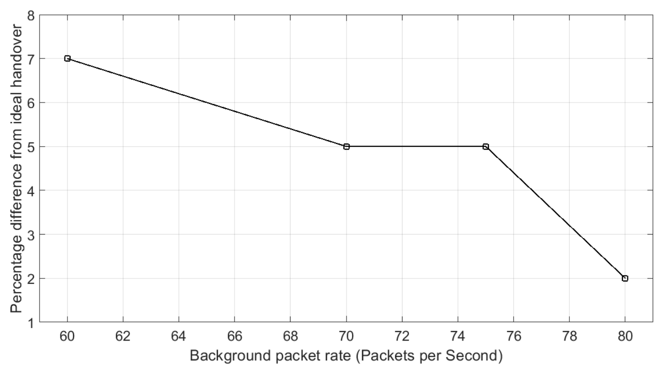
In this research, the capabilities of SDN are employed in a wireless setting of a smart city scenario where several WiFi networks provide overlapped coverage to smart city nodes. These nodes support delay-critical smart city applications or services. An algorithm is developed to run over the SDN controller to redistribute the WiFi nodes over the APs, covering their deployment area, based on the AP load and the inter-packet delay. The algorithm does not assume any WiFi nodes' special capabilities, such as running a software agent or supporting IEEE 802.11k/v. The SDN controller seamlessly



(a) E2E delay before and after handover of clients



(b) A comparison of e2e delay and interarrival time at the SDN controller.



(c) Percentage difference between ideal handover and algorithm decision.

Fig. 5. Algorithm performance with different background traffic loads.

performs the handover of WiFi nodes and continuously monitors the network performance in case the number of smart city nodes changes or the background traffic load increases. Extensive laboratory experiments show that the algorithm is efficient in reducing the end-to-end packet delivery delay for the WiFi nodes under study.

VIII. Acknowledgments

This research is supported by the UAE University UPAR grant number 12N008 and College of Engineering.

References

[1] C. V. N. Index, "Forecast and trends, 2017–2022 white paper," *Cisco: San Jose, CA, USA*, 2019.

[2] S. Seneviratne, Y. Hu, T. Nguyen, G. Lan, S. Khalifa, K. Thilakarathna, M. Hassan, and A. Seneviratne, "A survey of wearable devices and challenges," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2573–2620, 2017.

[3] V. Burger, M. Seufert, F. Kaup, M. Wichtlhuber, D. Hausheer, and P. Tran-Gia, "Impact of WiFi offloading on video streaming QoE in urban environments," in *2015 IEEE International Conference on Communication Workshop (ICCW)*. IEEE, 2015, pp. 1717–1722.

[4] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet of Things journal*, vol. 1, no. 1, pp. 22–32, 2014.

[5] E. Z. Tragos, V. Angelakis, A. Fragkiadakis, D. Gundlegard, C.-S. Nechifor, G. Oikonomou, H. C. Pöhls, and A. Gavras, "Enabling reliable and secure iot-based smart city applications," in *2014 IEEE International Conference on Pervasive Computing and Communication Workshops (PERCOM WORKSHOPS)*. IEEE, 2014, pp. 111–116.

[6] T. Pflanzner, K. Z. Leszko, and A. Kertész, "Summon: Gathering smart city data to support iot-fog-cloud simulations," in *2018 Third International Conference on Fog and Mobile Edge Computing (FMEC)*. IEEE, 2018, pp. 71–78.

[7] M. M. Islam, M. A. Razzaque, M. M. Hassan, W. N. Ismail, and B. Song, "Mobile cloud-based big healthcare data processing in smart cities," *IEEE Access*, vol. 5, pp. 11 887–11 899, 2017.

[8] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2014.

[9] "IEEE Standard for Local and metropolitan area networks–Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," *IEEE Std 802.11-2012 (Revision of IEEE Std 802.11-2007)*, pp. 1–2793, 2012.

[10] L. Yen, T. Yeh, and K. Chi, "Load Balancing in IEEE 802.11 Networks," *IEEE Internet Computing*, vol. 13, no. 1, pp. 56–64, 2009.

[11] "IEEE Standard for – Local and metropolitan area networks– Specific requirements– Part 11: Wireless LAN Medium Access Control (MAC)and Physical Layer (PHY) Specifications Amendment 1: Radio Resource Measurement of Wireless LANs," *IEEE Std 802.11k-2008 (Amendment to IEEE Std 802.11-2007)*, pp. 1–244, 2008.

[12] "IEEE Standard for– Local and metropolitan area networks– Specific requirements– Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications Amendment 8: IEEE 802.11 Wireless Network Management," *IEEE Std 802.11v-2011 (Amendment to IEEE Std 802.11-2007*, pp. 1–433, 2011.

[13] F. Xu, C. C. Tan, Q. Li, G. Yan, and J. Wu, "Designing a practical access point association protocol," in *2010 Proceedings IEEE INFOCOM*. IEEE, 2010, pp. 1–9.

[14] H. Gong, K. Nahm, and J. Kim, "Distributed fair access point selection for multi-rate ieee 802.11 wlans," in *2008 5th IEEE Consumer Communications and Networking Conference*, 2008, pp. 528–532.

[15] A. J. Nicholson, Y. Chawathe, M. Y. Chen, B. D. Noble, and D. Wetherall, "Improved access point selection," in *Proceedings of the 4th international conference on Mobile systems, applications and services*, 2006, pp. 233–245.

[16] M. Carrascosa and B. Bellalta, "Decentralized ap selection using multi-armed bandits: Opportunistic ϵ-greedy with stickiness," in *2019 IEEE Symposium on Computers and Communications (ISCC)*, 2019, pp. 1–7.

[17] T. Adame, M. Carrascosa, B. Bellalta, I. Pretel, and I. Etxebarria, "Channel load aware AP / Extender selection in Home WiFi networks using IEEE 802.11k/v," *IEEE Access*, pp. 1–1, 2021.

[18] A. Raschellà, F. Bouhafs, M. Seyedebrahimi, M. Mackay, and Q. Shi, "A centralized framework for smart access point selection based on the fittingness factor," in *2016 23rd International Conference on Telecommunications (ICT)*. IEEE, 2016, pp. 1–5.

[19] S. Manzoor, C. Zhang, X. Hei, and W. Cheng, "Understanding traffic load in software defined wifi networks for healthcare," in *2019 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*. IEEE, 2019, pp. 1–2.

[20] A. Abdrabou and W. Zhuang, "Stochastic delay guarantees and statistical call admission control for IEEE 802.11 single-hop ad hoc networks," *IEEE Transactions on Wireless Communications*, vol. 7, no. 10, pp. 3972–3981, 2008.

[21] "RYU SDN Framework." [Online]. Available: https://book.ryu-sdn.org/en/html/

[22] E. Coronado, S. N. Khan, and R. Riggio, "5G-EmPOWER: A software-defined networking platform for 5G radio access networks," *IEEE Transactions on Network and Service Management*, vol. 16, no. 2, pp. 715–728, 2019.